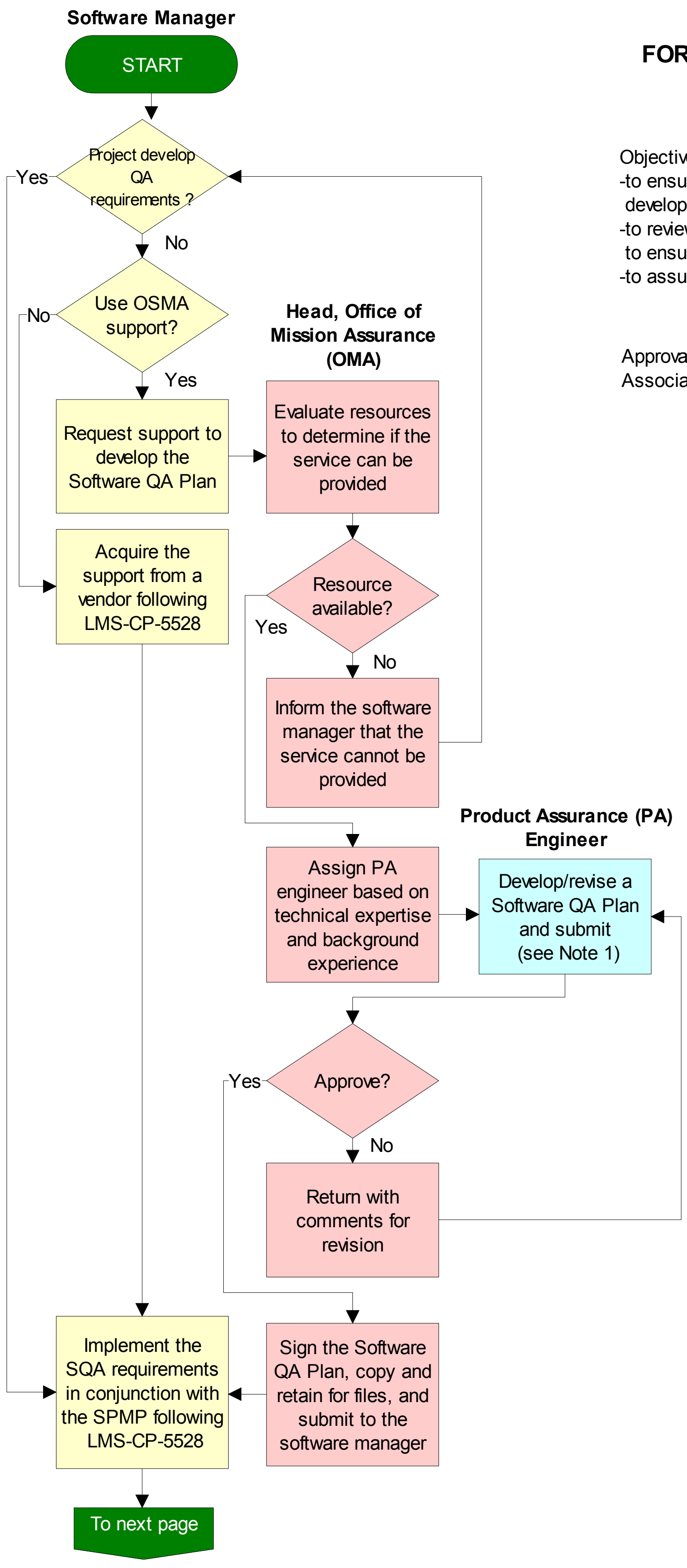
QUALITY ASSURANCE (QA) FOR SOFTWARE DEVELOPMENT AND ACQUISITION


Software Manager
START
Project develop QA requirements ?
Yes
No
Use OSMA support?
No
Yes
Request support to develop the Software QA Plan
Acquire the support from a vendor following LMS-CP-5528
Head, Office of Mission Assurance (OMA)
Evaluate resources to determine if the service can be provided
Resource available?
Yes
No
Inform the software manager that the service cannot be provided
Assign PA engineer based on technical expertise and background experience
Product Assurance (PA) Engineer
Develop/revise a Software QA Plan and submit (see Note 1)
Approve?
Yes
No
Return with comments for revision
Sign the Software QA Plan, copy and retain for files, and submit to the software manager
Implement the SQA requirements in conjunction with the SPMP following LMS-CP-5528
To next page

Objectives:
-to ensure that software is correctly classified prior to commencing development
-to review software development plans, products and related process to ensure that they meet the predefined set of guidelines and standards
-to assure all software developed meets stated requirements

Approval Original signed by Lana M. Couch
Associate Director for Business Management

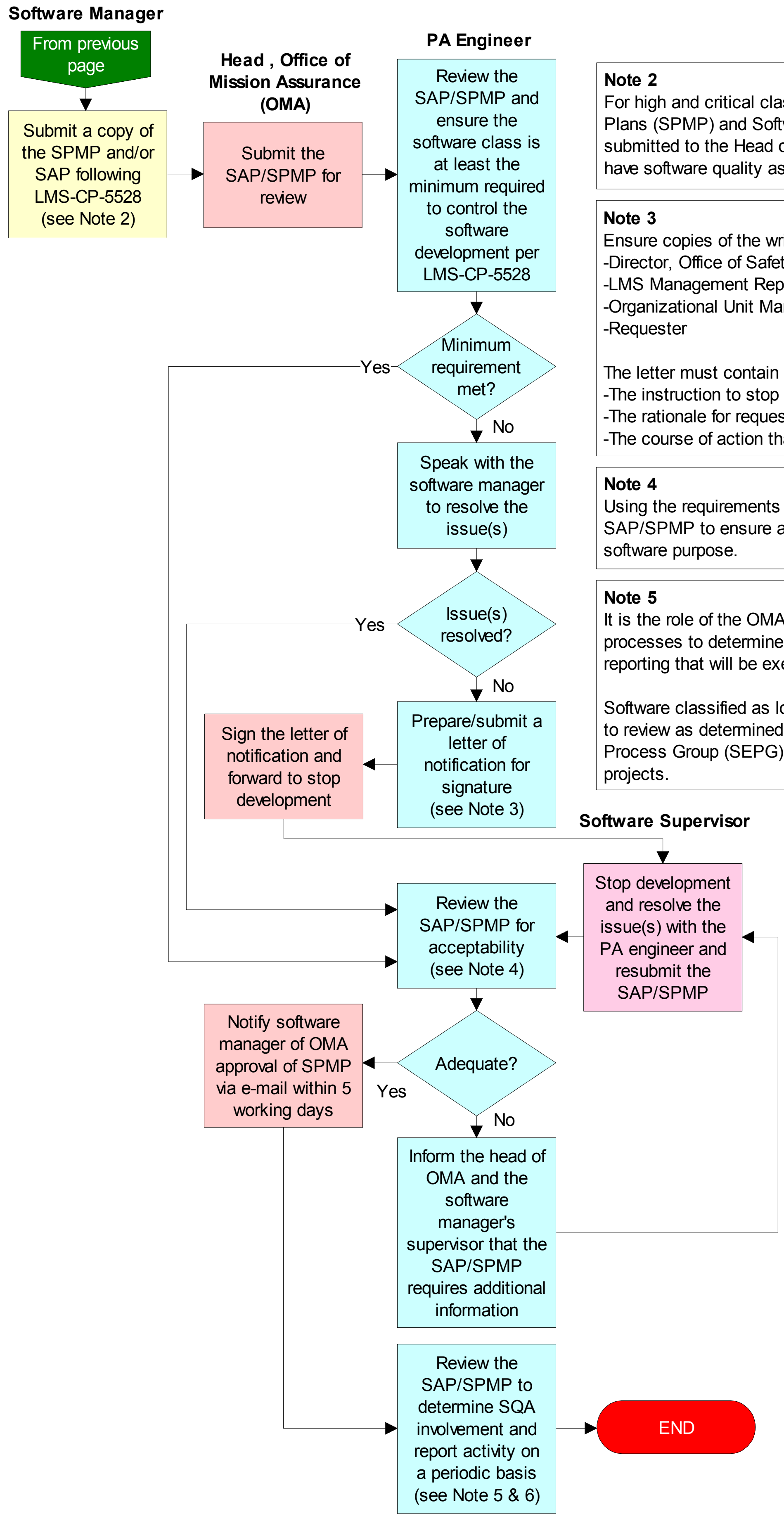
General Information

The following records are generated by this procedure and should be maintained in accordance with LMS-CP-2707: Software Plan

Note 1
Use the template, and checklists located at the Appendices below to develop the Software QA Plan:

Appendix A - Software QA Guidance
Appendix B - Sample Software QA Plan Template
Appendix C - Software QA Checklists
Appendix D - Software QA Tools

Develop software QA requirements with the Software Manager once the review of the Software Acquisition Plan (SAP) or Software Project Management Plan (SPMP) is completed in accordance with LMS-CP-5528.


Software Manager
From previous page
Submit a copy of the SPMP and/or SAP following LMS-CP-5528 (see Note 2)
Head , Office of Mission Assurance (OMA)
Submit the SAP/SPMP for review
PA Engineer
Review the SAP/SPMP and ensure the software class is at least the minimum required to control the software development per LMS-CP-5528
Minimum requirement met?
Yes
No
Speak with the software manager to resolve the issue(s)
Issue(s) resolved?
Yes
No
Prepare/submit a letter of notification for signature (see Note 3)
Sign the letter of notification and forward to stop development
Software Supervisor
Stop development and resolve the issue(s) with the PA engineer and resubmit the SAP/SPMP
Review the SAP/SPMP for acceptability (see Note 4)
Adequate?
Yes
No
Notify software manager of OMA approval of SPMP via e-mail within 5 working days
Inform the head of OMA and the software manager's supervisor that the SAP/SPMP requires additional information
Review the SAP/SPMP to determine SQA involvement and report activity on a periodic basis (see Note 5 & 6)
END

Note 2
For high and critical class software, Software Project Management Plans (SPMP) and Software Acquisition Plans (SAP) must be submitted to the Head of the OMA, including those that do not have software quality assurance requirements.

Note 3
Ensure copies of the written notification are sent to:
-Director, Office of Safety and Mission Assurance
-LMS Management Representative
-Organizational Unit Manager of the Software Manager
-Requester

The letter must contain the following information:
-The instruction to stop development
-The rationale for requesting reevaluation
-The course of action that will be taken to resolve the dispute

Note 4
Using the requirements given in LMS-CP-5528, review the SAP/SPMP to ensure adequate coverage for the given class and software purpose.

Note 5
It is the role of the OMA to review all SAP/SPMP and associated processes to determine its level of on- and off-site monitoring and reporting that will be exercised during the development life cycle

Software classified as low-control per LMS-CP-5528 will be subject to review as determined by the OMA. Software Engineering Process Group (SEPG) Data base will be used to select low-control projects.

Note 6
It is the responsibility of the PA Engineer to compile a monthly report on projects reviewed. The report must contain:
-Total number of Plans reviewed
-Breakdown of projects reviewed by software class
-Number of incorrect classifications
-Adequacy of Plan content
-Results of on- and off-site monitoring

Appendix A: Software Quality Assurance Guidance

1. Introduction

Software quality assurance is the process of reviewing all software development products and related processes to ensure that they meet a predefined set of requirements and standards. This involves selecting standards and additional requirements for the following phases, including software output requirements, design, code implementation, test, documentation, and verifying that the standards and requirements are met.

The three mutually supportive activities involved in the software life cycle are management, engineering, and assurance. Software management is the set of activities involved in planning, controlling, and directing the software project. Software engineering is the set of activities that analyzes requirements, develops designs, develops code, and structures databases. Software assurance is responsible for verifying that the management and engineering efforts put forth result in a product that meets all of its requirements.

Software assurance is the planned and systematic set of activities that ensures that software processes and products conform to requirements, standards, and procedures. "Processes" include all of the activities involved in designing, developing, enhancing, and maintaining software; "products" include the software, associated data, its documentation, and all supporting and reporting paperwork.

Software Quality Assurance (SQA) is the sub-discipline of software assurance that is defined as a planned and systematic approach to the evaluation of the quality of and adherence to software product standards, processes, and procedures. SQA includes the process of ensuring that standards and procedures are established and are followed throughout the software acquisition life cycle. The objective of quality assurance is to ensure that each software product fulfills its specific intent. This is accomplished in the review/survey process of all project deliverables.

Software quality assurance shall be made an integral part of the software development activities by initiating it early in the project development process. The standards, requirements, and conventions that establish the ground rules for the development process are identified in a Software Assurance Plan, in some cases, part of the Product Assurance Plan or the Software Management Plan. This plan is published and distributed to all development team members, and frequent reviews are made to eliminate doubts regarding the standards and processes to be used throughout the project life cycle. Software quality assurance is directed toward the format and content of the software products and processes, and the standards used to develop them. Ensuring that products and processes represent and meet the project requirements is addressed through analysis activities performed by the Project SQA representative.

2. Planning

Software quality assurance requirements are defined in LMS-CP-5528.

3. Software Assurance Activities Requirements

The level of detail required for the software assurance activity depends on the specific project. The primary objective is to tailor the level of detail to individual software products. Judgement must be applied when determining the extent of assurance activities to be conducted for each software deliverable. Tradeoffs have to be considered regarding the cost of the software assurance activities and the value added by the activity in relation to the specific software product.

4. Software Quality Assurance Plans

The Software Quality Assurance (SQA) Plan specifies the assurance procedures to be applied during each phase of the software development life cycle and assigns responsibilities for software quality assurance activities. SQA Plans can be stand alone, included as part of a Product Assurance Plan (PAP), or included in a Software Project Management Plan (SPMP). The project manager and software manager are responsible for ensuring that the plan is tailored to each project and that it contains the following information:

Standards and conventions to be followed by the development team

Formats for document deliverables

Checklists for evaluating standards compliance and product completeness

Deviations from the policies and procedures and the rationale for deviation

Specific software products to be monitored by quality assurance procedures

Methodologies and tools to be used for each product

Quality assurance milestones based upon project milestones and product deliverables

Key personnel responsible for software quality assurance for the project and those who will review and audit related products.

Appendix B contains a sample SQA Plan template for developing project software quality assurance plans. The following documents also contain information for developing a SQA Plan: IEEE/EIA Standard 12207.1-1997 and IEEE 730-1989 Software Quality Assurance Plans.

5. Software Quality Assurance Methods

The primary methods used in the software quality assurance process are:

Audits
Reporting and control
Life cycle reviews
Surveys
Spot checks / inspections
Code analysis / walkthroughs

Appendix B: Sample Software Quality Assurance Plan Template

The following pages contain a detailed Software Quality Assurance Plan (SQAP) template that shall be used as an example in preparing a SQAP for a specific project. In preparing a specific SQAP, this template SQAP shall be tailored by the acquirer (e.g., LaRC program/project manager) in accordance with the software project and the software class as defined in LMS-CP 5528. This SQAP is starting point, not the final answer, in planning software assurance activities.

To facilitate the conversion of this template SQAP into a specific SQAP, there are several global replacements that can be made to speed the tailoring. The items shown below in (Bold Italic) parentheses are the subjects for global replacement and provide some guidance on how to define what the replacement text should be.

(Provider) The name of the company or organization that is supplying/developing the software and is responsible for the quality of the software product.

(Acquirer) The name of the organization that is acquiring/purchasing the software products for use by their organization.

(XYZ Project) The name of the project for which the plan is written.

(XYZ Project Manager) The individual managing the XYZ Project. This individual's responsibilities may encompass direction of hardware engineering, systems engineering, and production tasks as well as software engineering tasks.

An italicized example is supplied for each section.

Software Quality Assurance Plan

for the

(XYZ Project)

[document date]

Submitted By:	____ Software Quality Assurance Engineer	____________ Date
Concurred By:	____ ***(XYZ Software Development Technical Manager)***	____________ Date
Approved By:	____ ***(XYZ Project Manager)***	____________ Date

National Aeronautics and Space Administration
Langley Research Center
Hampton, Virginia

Table of Contents

APPENDICES

FIGURES

FORMS

1. INTRODUCTION

This section should introduce the Software Quality Assurance Plan (SQAP). The SQAP should describe the organization and procedures to be used to ensure that the software to be delivered complies with the system requirements as well as technical standards.

The following italicized text offers an example for this section:

This Software Quality Assurance Plan describes the organization and procedures to be used by the National Aeronautics and Space Administration (NASA) Langley Research Center (LaRC) to ensure that the software to be delivered for the **(XYZ Project)** complies with the application requirements, as well as the technical standards set by the NPD 2820.1 NASA Software Policies, the NASA-STD-2201-93 Software Assurance Standard, and the NASA-STD-2100-91 Software Documentation Standards.

1.1 Identification

This section should identify this plan and the XYZ project.

The following italicized text offers an example for this section:

*This document is identified as the NASA LaRC Software Quality Assurance Plan (SQAP) for the **(XYZ Project)**. This plan establishes the policies, standards, procedures, and practices for software assurance of all computer software for applications developed for or by LaRC. Moreover, this plan establishes a standards compliance perspective for LaRC.*
(About 2 to 4 paragraphs - should identify the XYZ project)

1.2 Purpose

This section should describe the purpose of the SQAP.

The following italicized text offers an example for this section:

The purpose of the SQAP is to ensure the quality of newly developed and modified software. This will be accomplished by the following software quality assurance tasks: Audits, Reporting and Control, Life Cycle Reviews, and Surveys which include Process Evaluations, Spot checks / Inspections, and Code Analysis / Walkthroughs.
(About 2 to 4 paragraphs - should state the purpose of the system and the functions of software for which this SQAP applies)

1.3 Scope

This section should describe the scope of the SQAP.

The following italicized text offers an example for this section:

*The scope of this document is to define the activities which will be carried out in support of software quality assurance for the **(XYZ Project)**.*
This document defines the actions that provide assurance that the software-related items delivered conform to their established and contracted technical requirements. Software Quality Assurance (SQA) will also ensure that defined standards, practices, procedures, and methods of the software development process are applied.
This plan contains 1) orientation material, 2) the organization and resources that will be used to accomplish the SQA activities, 3) the planned schedule of SQA activities, 4) the planned SQA activities and the tools/procedures that support those activities, and 5) the quality assurance records that will be kept.
This plan conforms to the requirements of NASA-STD-2201-93, ISO 9000-3, and SEI-TR-25.

1.4 Applicable Documents

This section should list all documents, especially those prepared for the XYZ Project, which affect or are affected by the SQA activities. This section should also present a bibliography of reference and supporting documents.

The following italicized text offers an example for this section:

The documents of the exact issue as shown below, form a part of this document to the extent described herein. In the event of a conflict between the documents referenced herein and the contents of this document, the contents of this document shall be considered a superseding requirement to the previous documents.

The following documents and publications provide information pertinent to the information in this document.

1. XYZ Program Plan
2. XYZ Program Technical Manager's Guidebook
3. XYZ Product Assurance Plan

The following list contains all related documents which provide supporting information to this SQAP.

1. Software Requirements Document
2. Software Standards and Procedures Manual
3. System/Software Support Plan
4. Software Process and Procedures Document

Other plans related to the quality of software development includes the **(XYZ Project)** Software Development Plan (SDP) which describes the standards, schedule and procedures to be followed by the development team, the **(XYZ Project)** Software Configuration Management Plan (SCMP) which describes how products of this project will be identified, baselined and controlled, and the ***(XYZ Project)*** *Software Test Plan and Procedures (STPP) which describes how products of this project will be tested.*

(The following standards should also be referenced.)

ISO 9000-3:1991	Quality management and quality assurance standards -- Part 3; Guidelines for the application of International Organization for Standardization (ISO) 9001 to the development, supply and maintenance of software.
IEEE/EIA Standard 12207.0-1996	Industry Implementation of International Standard ISO/IEC 12207: 1995, Standard for Information Technology – Software Life Cycle Processes.
IEEE/EIA Standard 12207.1-1997	Industry Implementation of International Standard ISO/IEC 12207: 1995, Standard for Information Technology – Software Life Cycle Processes – Life Cycle Data.
NASA-STD-2201-93	NASA Technical Standards Division, Software Assurance Standard

2. ORGANIZATION AND MANAGEMENT

This section should discuss the Project organization in terms of structure, key personnel, and resources.

2.1 Organization

This section should describe the relationship between the SQA organizational entities in the project and refers to an organization structure chart. The chart indicates the position of the project management organization and details the organizational structure.

The following italicized text offers an example for this section:

The SQA representative is separate from the software developers and reports directly to the Project Manager (PM). The SQA representative is obligated to fully advise the PM of the status of the software and the accompanying documentation.

Figure 2.1-1 ***(insert an organization chart with named individuals)*** *presents the reporting chain.*

The Project is responsible for the fulfillment of, and for ensuring compliance with, the software quality assurance activities defined in this document. The Project is responsible for ensuring that processes are applied as intended and that products are of high quality. The SQA representative provides senior ***(Provider****) management with an independent evaluation of the* ***(XYZ Project****), and it provides a direct reporting line to the senior management of* ***(Provider****) to resolve problems.*

2.2 Resources

This section should describe the specific resources necessary to implement the SQAP.

Included in resources are such items as:

- Personal Computers
- Clerical support
- PC word processing software
- Performance Measurement System

2.2.1 SQA Equipment, Facilities and Software

This section should describe the SQA equipment, facilities and software used for the XYZ project.

The following italicized text offers an example for this section:

The SQA Representative shall use the following general purpose Personal Computer (PC) software for creating reports and documents, tracking SQA activities, and other activities defined in this plan:

(List of database management, spreadsheet, project tracking, communications, and networking software that will be used)

The following tools will be used in the evaluation and tracking of software products:

(List of the comparison, analysis, testing, standards auditor, problem report tracking, configuration management, and similar tools that shall be used)

2.2.2 Customer Furnished Equipment, Facilities and Software

This section should describe the customer-furnished equipment, facilities and software used by the SQA Representative for the XYZ project.

The following italicized text offers an example for this section:

(If customer furnished equipment, software, or services are required to support the development, evaluation, testing, or installation of the software products being tested they should be listed here. If none are needed the following sentence should be included:)

There are no requirements for using customer-furnished facilities in executing this Software Quality Assurance Plan.

2.2.3 Personnel

This section should list the key personnel to be involved in the SQA activities by title and minimum qualifications for the position.

The following italicized text offers an example for this section:

The Project shall select an SQA representative experienced in the Software Life Cycle.

The SQA representative is responsible for evaluating the quality (usability, reliability, etc.) of the software being developed. They have a background in Software Engineering principles with experience in developing software. They understand the needs of the end user in order to judge how well the product meets real-world needs.

2.2.4 Other Resources

This section should describe the resources used by the SQA Representative for the XYZ project.

The following italicized text offers an example for this section:

Post-development phases of the ***(XYZ Project)*** *shall require additional resources. These resources are:*

(List of facilities needed to continue software quality assurance activities in support of post-development test and evaluation and in customer support such as network-capable software problem report tracking tools)

2.3 Schedule

This section should present the major milestones of the software development schedule for the XYZ project and corresponding SQA activities.

The following italicized text offers an example for this section:

Figure X presents the major milestones of the software development schedule for the ***(XYZ Project)*** *and the corresponding SQA activities that shall be performed during this development. The SQA activities shall be performed continuously during the* ***(XYZ Project)****.*

2.4 Cost

This section should describe any special funding or cost items associated with the SQA activities that will be provided by the project, i.e., independent verification and validation by contracting.

The following italicized text offers an example for this section:

The software quality assurance activities for the ***(XYZ Project)*** *shall be processed by competitive placement with a budget of (insert dollar amount). This activity shall include independent verification and validation depicted in the contract statement of work.*

3. GENERAL SOFTWARE QUALITY ASSURANCE ACTIVITIES

This section should discuss quality assurance procedures for:

- Audits
- Process Evaluations
- Reporting and Control
- Life Cycle Reviews
- Surveys
  - Spot checks / Inspections
  - Code Analysis / Walkthroughs

3.1 Audits

This section should describe the SQA audit procedures to be used in order to ensure that the software development process is proceeding in accordance with the SDP and includes a requirement for a schedule which will be maintained in the SDP for quality audits.

The following italicized text offers an example for this section:

The SQA Representative shall audit the software quality assurance activities performed for the ***(XYZ Project)*** *according to the procedures described in this section.*

Throughout the life cycle of the project, the SQA Representative shall perform audits to determine whether software procedures, standards, and practices have been identified and are being properly implemented. Audits also identify those areas in which additional controls and standards are required to ensure the quality of the software product. An audit may be a stand-alone activity or incorporated into a review.

3.1.1 Audit Planning

This section should describe the audit planning process in terms of schedules and procedures.

The following italicized text offers an example for this section:

The development and maintenance of schedules for internal audits shall be the responsibility of the SQA Representative. These schedules shall be based on the following:

- *Software Life Cycle phases*
- *Specific software products of each phase*
- *Previous audits results*

Prior to the start of the audit, the following activities shall be completed:

- *Review of the standards, practices, requirements, approved procedures, and directives of the area or function to be audited*
- *Preparation of checklists as needed*
- *Establishment of the audit team and the functions, procedures, and data to be audited.*

3.1.2 Audit Areas

This section should identify the activities to be audited by the selected SQA representative. The project specific areas to be audited should come from Table 2-4 in the Software Program Assurance LHB.

The following italicized text offers an example for this section:

Audits by the SQA representative shall be conducted of the following areas:

- *System Definition*
- *Software Design and Development*
- *Configuration Management*
- *Programming Library System*
- *Software System Test*
- *Sustaining Engineering Support*

The SQA representative shall use the checklists in Section 7 of this plan for the assessment of the SQA activities for the ***(XYZ Project).***

3.1.3 Audit Execution

This section should describe the approach used in executing an audit.

The following italicized text offers an example for this section:

The SQA representative may request support from the engineering organization responsible for the area to be audited. This support may be in the form of personnel, records, files, and procedures. Examination during an audit may include both processes and products.

The results and findings of an audit shall be documented in a report following the guidelines outlined in Section 3.3 of this plan.

3.2 Evaluating Processes and Procedures

This section should describe the activities that will occur when evaluating the software processes and procedures.

The following italicized text offers an example for this section:

The following software processes will be evaluated by an assessment panel selected by the Project.

- *Requirement Management*
  - *System requirements allocated to software are controlled to establish a baseline for software engineering and management use.*
  - *Software plans, products, and activities are kept consistent with the system requirements allocated to software.*

- *Software Project Planning*
  - *Software estimates are documented for use in planning and tracking the software project.*
  - *Software project activities and commitments are planned and documented.*
  - *Affected groups and individuals agree to their commitments related to the software project.*

- *Software Project Tracking and Oversight*
  - *Actual results and performances are tracked against the software plans.*
  - *Corrective actions are taken and managed to closure when actual results and performance deviate significantly from the software plans.*
  - *Changes to software commitments are agreed to by the affected groups and individuals.*

- *Software Product Engineering*
  - *The software engineering tasks are defined, integrated, and consistently performed to produce the software.*
  - *Software work products are kept consistent with each other*

- *Software Quality Assurance*
  - *Software quality assurance activities are planned.*
  - *Adherence of software products and activities to the applicable standards, procedures, and requirements is verified objectively*
  - *Affected groups and individuals are informed of software quality assurance activities and results*
    - *Noncompliance issues that cannot be resolved within the software project are*
    - *Addressed by senior management.*

- *Software Configuration Management*
  - *Software configuration management activities are planned.*
  - *Selected software work products are identified, controlled, and available.*
  - *Changes to identified software work products are controlled.*
    - *Affected groups and individuals are informed of the status and content of software baselines.*
- Software Contract Management
  - The prime contractor selects qualified software subcontractors.
  - The prime contractor and the software subcontractor agree to their commitments to each other.
  - The prime contractor and the software subcontractor maintain ongoing communications.
  - The prime contractor tracks the software subcontractor's actual results and performance against its commitments.

3.3 Reporting and Control

This section should describe the SQA reporting and control procedures, including how SQA information will be made available to project management and how inadequacies, discrepancies, and deficiencies, as well as proposed improvements, will be highlighted to management.

The following italicized text offers an example for this section:

When an anomalous condition is found in the development or maintenance process, the SQA representative will document the problem on the System/Software Problem Report (SPR). Appendix A of this plan contains a copy of the SPR form with instructions. These reports are the end result of reviews, audits, and tests. The SPR will be tracked from its inception to resolution by the SQA representative.

The SQA representative has the responsibility of establishing and maintaining a project SPR file. Each file will contain the SPR, inclusive of checklists, notes and procedures, the corrective action replies, and any associated correspondence.

During reviews, audits, and testing, a SPR will be written against questionable items. These SPRs will be answered by the appropriate engineering organization within a 30-day period. The SQA representative has the responsibility to monitor the SPR process to ensure that all SPRs are answered correctly and completely.

3.3.1 Problem Reporting and Corrective Action

This section should describe the SQA activities for evaluating the problem reporting and corrective action processes and procedures.

The following italicized text offers an example for this section:

The SQA representative will evaluate the corrective action system for handling SPRs according to the procedures described in this section. The objective of the corrective action system is to provide a systematic method of resolving identified problems. Identification of problems may be based on:

a. *contractual, company, departmental, or program procedural nonconformance;*
b. *unsatisfactory quality trends discovered in internal reviews and walkthroughs;*
c. *functional deficiencies and performance problems traceable to software errors,*
d. *unsatisfactory supplier or subcontractor performance;*
e. *and inconsistencies, errors, and omissions in specifications and documents.*

Each time a software failure is observed or a software error is discovered, a record shall be made to make certain that corrective actions are taken to remove the error on a form, called the SPR.

Action items requiring corrective action may result from reviews and walkthroughs. When corrective action is taken and verified by the review/walkthrough leader, the completed action item list is sent to the PM. Regardless of the source of the problem, the SQA representative shall conduct a follow-up inspection of the corrected documents or code to verify correction of the original deficiency.

The following characteristics are essential of the corrective action system:

a. *correct, complete, and accurate problem reporting;*
b. *careful analysis of the problems reported;*
c. *proper classification of the problems as to category and severity;*
d. *assignment of responsibility for correcting the problem;*
e. *determination of effective actions to be taken to remedy the problem;*
f. *regular, thorough analysis of trends;*
g. *effective, implementable recommendations resulting from trend analysis;*
h. *proper authorization of steps to correct problems;*

i. *complete records of documented actions taken;*
j. *reevaluation of corrective actions after being taken;*
k. *tracking corrective action progress and closing out completed actions; and*
l. *customer visibility into the corrective action process.*

Table 4 in Section 7 contains a checklist for the SQA evaluation of the operation and effectiveness of the corrective action system.

3.3.2 Quality Assurance Records

This section should describe the maintenance of the quality assurance records, the SQA files and the metrics used by the SQA representative to track the SQA activities for the XYZ project.

The following italicized text offers an example for this section:

Records of the SQA activities described in this plan will be kept in the ***(XYZ Project)*** *files. These records may be inspected at any time.*

The ***(XYZ Project)*** *files may include the following records:*

- *Annotated copies of* ***(XYZ Project)*** *documents that have been the subject of SQA Inspections, Audits and Reviews*
- *Data and Analysis Graphics for Software Metrics*
- *Completed Checklists*
- *Internal Review Reports*
- *Approval Record Sheets*
- *Monthly SQA Reports*
- *Document Review Reports*
- *Test Observation Records*
- *Problem Reports and Corrective Action Records*
- *Issue Tracking Forms*

To assist in making and evaluating the software development products and processes, basic software metrics shall be used. These basic metrics are:

1) *Product size, of the total software product and of each major component of that product, measured in* ***(Indicate whether Lines of Code, or Function Points or both will be used as the size measure)****.*
2) *Person-months for the total project and by project phase and for major software components.*
3) *Schedule time for the total project and by project phase.*
4) *Number of software errors made and discovered in each project development phase.*
5) *Number of software errors discovered after completion of the software development.*
6) *Software requirements volatility (i.e., the percent of software errors that have been modified or added after the end of the software requirements definition phase).*
7) *Number of tests successfully executed as a percentage of the total number of tests planned.*
8) *The total number software problem reports written compared with the number of software problem reports that have been closed.*

The SQA representative will support the collection of data for these metrics and shall prepare reports documenting the conclusions of the analysis of the metrics data. Graphic plots of the above metric values versus time shall be used by the SQA representative to reveal favorable or unfavorable trends. The SQA representative shall also contrast the metric values obtained during the ***(XYZ Project)*** *with values obtained from past projects.*

3.4 Life Cycle Reviews

This section should describe the life cycle reviews.

The following italicized text offers an example for this section:

Life cycle reviews are formalized activities which maintain visibility into the process of software development. These reviews are scheduled at the beginning of the project and represent the necessary milestones in the development process. The responsible *organizations initiating the review shall ensure proper notification of impending reviews. Section 4 of this plan contains detailed SQA activities for each life cycle review phase.*

3.5 Surveys

3.5.1 Spot Checks / Inspections

This section should describe the review process, for specification reviews, project documentation, and code reviews.

The following italicized text offers an example for this section:

Spot checks and inspections are short-notice reviews of the software process and product samples to monitor standards compliance. The SQA representative shall perform spot checks with each software developer in the first few weeks of each new product development activity and provides the software developers with specific feedback on standards compliance. The SQA role is to observe, participate as needed, and verify that the spot checks and inspections were properly conducted and documented.

3.5.2 Code Analysis / Walkthroughs

This section should describe the SQA activities performed during the project code analysis/walkthroughs.

The following italicized text offers an example for this section:

It will be necessary to conduct internal code analysis/walkthroughs prior to the reviews, since reviews are "after-the-fact" evaluations. The SQA representative shall be present to lend support in interpretation of standards and contractual aspects as they apply to the subject of the code analysis/walkthrough.

*A walkthrough is the procedure used by the leader of a review meeting. The walkthrough is his means of leading the audience through the product with the goal of uncovering problems, inconsistencies, etc., which would be otherwise overlooked. The walkthrough procedure will be used whenever **(XYZ Project)** software development products or processes are scheduled to undergo internal review, according to the **(XYZ Project)** schedule. Thus, the topic of a walkthrough can be the top-level design of the software, the detailed design, the software user manual, etc.*

Code Analysis verifies that the coded program correctly implements the verified design requirements. The techniques used in performance of code analysis mirror those used in design analysis.

The SQA role is to observe, participate as needed, and verify that the analyses and walkthroughs were properly conducted and documented.

4. LIFE CYCLE PHASE SOFTWARE QUALITY ASSURANCE ACTIVITIES

4. 1 Software Concept and Initiation Phase

This section should describe the SQA activities that will occur during this phase.

The following italicized text offers an example for this section:

*The SQA representative will review the Software Management Plan to verify that it complies with all **(Acquirer)** requirements and describes a practical and effective software development process. In particular, the SQA representative will confirm that adequate provisions have been made for error prevention and early error discovery procedures, techniques and tools.*

The SQA representative shall review the allocation of user needs and system requirements between hardware and software segments of the system to confirm that no requirement has remained unallocated and to verify that the software can meet the requirements allocated to it. In conducting this review the SQA representative shall use the checklist presented in Table 5 of Section 7.

4.2 Software Requirements Phase

This section should describe the SQA activities that will occur during this phase.

The following italicized text offers an example for this section:

The SQA representative shall review the software requirements prepared by the development team to make certain they are complete, approved and consistent and are a suitable foundation for the subsequent design and test activities. In analyzing the quality of requirements, special attention will be paid to the "testability" of each requirement. That is, if there is no straightforward way to test or inspect whether or not a requirement is met, then the statement of the requirement is deficient.

The SQA representative will review the software requirements document using the checklist presented in Table 6 in Section 7. This document will be checked for compliance with its applicable document preparation standard to verify that it has the requisite format and content.

*The SQA representative will also assist the **(XYZ Project Manager)** by verifying that the necessary documentation and preparation steps leading up to the Software Requirements Review (SRR) have been followed. The requirements definition phase will end with the Software Requirements Review.*

4.3 Software Design Phase

This section should describe the SQA activities that will occur during this phase.

The following italicized text offers an example for this section:

The SQA representative will assist the **(XYZ Project Manager)** by verifying that the necessary documentation and preparation steps leading up to the Preliminary Design Review (PDR) have been followed. The first portion of the design phase will end with the Preliminary Design Review.

*The SQA representative will also assist the **(XYZ Project Manager)** by verifying that the necessary documentation and preparation steps leading up to the Critical Design Review (CDR) have been followed. The final portion of the design phase will end with the Critical Design Review.*

The SQA representative will review the design documents presented in Table 7 in Section 7. These documents will be examined to make certain they are in their required form and have the required content.

4.3.1 Project Documentation Review

This section should describe the SQA procedures to be used in reviewing documentation for conformance with the project and is to include a schedule requirement of the design documentation reviews.

The following italicized text offers an example for this section:

Documentation throughout life cycle of the project must be inspected and checked to ensure that it is correct. The following criteria shall be used:

- *Compliance to documentation standards*
- *All necessary information has been included*
- *Text is clear and unambiguous*
- *Adequate traceability and cross-referencing has been provided and is maintained*

Each phase of the life cycle generates different documents which need to be inspected. These documents shall be subjected to a SQA inspection one week prior to the formal review of which it will be a part.

The following is a list of documentation:

- *Project Management Plan*
- *Functional Requirements Document*
- *Software Development Plan*
- *Software Preliminary Design Specification*
- *Software Detailed Design Specification*
- *User and Operations Guide*
- *Training Plan and Procedures*
- *Test Plan and Procedures*
- *Site Installation Plan*

4.4 Software Implementation Phase

This section should describe the SQA procedures to be used in reviewing the actual software code for compliance with the applicable coding standards and is to include a schedule of the code quality reviews.

The following italicized text offers an example for this section:

*The source code is the programming language implementation of the software design. The SQA representative shall confirm that the implemented code is traceable to the functions specified in the design documents produced in earlier project phases. The SQA representative will also confirm that the source code follows coding standards defined in **(XYZ Project)** Software Development Plan.*

One month prior to submitting the code for software integration, the software code is to be subjected to SQA for review.

Table 8 in Section 7 presents the SQA checklist that shall be used for source code reading.

4.5 Software Integration and Test Phase

This section should describe the SQA activities that will occur during this phase.

The following italicized text offers an example for this section:

The SQA representative shall review the software test plan to ensure that it describes a software test activity that thoroughly exercises the software implementation and provides evidence that all software requirements have been satisfied. This review shall be done using the checklists presented in Table 9 in Section 7.
The SQA representative shall review the software test procedures for the ***(XYZ Project)*** *to ensure that they are consistent with the software test plan and describe effective and comprehensive tests. A major criterion in evaluating these test procedures is to make certain that the descriptions of the test are sufficient for a person other than the test procedure author to conduct the test. The checklist for this test procedure review is presented in Table 10 of Section 7.*

The SQA representative shall witness selected software test and review the results of many additional tests that it does not witness. The purpose of this test witnessing and the test results review is to confirm that the tests were run as described in the test procedures and that the results are acceptable. This review of the test conduct shall use the checklist presented in Table 11 of Section 7.

4.5.1 Testing

This section should describe testing SQA processes, including plans, tests, and procedures.

The following italicized text offers an example for this section:

Testing is a quality activity performed by a separate group and reviewed by the SQA representative. Its primary concern is the controlled exercise of the program code using sample input cases with the objective of exposing errors. This process begins with the smallest unit of the system, continues up to, and includes installation.

In order to accomplish this task, it is necessary to generate a Software Test Plan and Procedures (STPP) document. This product must be reviewed by the responsible engineering organization manager and SQA representative to ensure their completeness according to standards.

4.5.1.1 Software Test Plan

This section should describe the SQA procedures to be used in reviewing the test plan.

The following italicized text offers an example for this section:

The test plan must be subjected to a SQA review prior to the PDR and the CDR. The test plan will contain test objectives, methodologies (from unit test to installation and checkout), description of the test environment, test description, delineation of the requirements verified, an evaluation plan, and a test schedule.

The test plan is begun in the SRR and revised as the system matures. This maturation process continues until CDR. One week prior to PDR and again one week prior to CDR this plan is submitted for a SQA inspection.

4.5.1.2 Software Test Specifications and Procedures

This section should describe the SQA procedures to be used in reviewing the test specifications and procedures.

The following italicized text offers an example for this section:

Software test specifications and procedures are step-by-step descriptions of the actions necessary to execute a particular test. Each test procedure shall contain the scope of the test, the applicable documents, organizational responsibilities, test environment, test data, criteria for acceptance, and the steps necessary to complete the test. Each individual step should be comprised of the user input and the expected result.

Formal test procedures shall be subjected to a SQA review one month prior to the test. The SQA representative will review the procedures to determine their completeness based on requirements and contractual agreement.

4.5.1.3 Software Test Performance

This section should describe the SQA procedures to be used in monitoring the software test activity.

The following italicized text offers an example for this section:

The SQA representative shall be present at all formal tests. It is the responsibility of the SQA representative to note any deviation from the accepted procedures. As the test is being conducted, the SQA representative will initial each step as it is completed. Hard copies of data (logs or copies of screen images) and output media (diskette or tape) will be baselined by the SQA representative.

4.5.1.4 Software Test Assurance Reports

This section should describe the SQA procedures to be used in reviewing SPRs.

The following italicized text offers an example for this section:

Upon completion of the formal test, a post test briefing will be conducted by the test team. This will occur within a one-week period in order to give time for any off-line analysis that must take place. During the meeting with the test personnel, programmers, customer, and the SQA representative, the results of the test will be reviewed. Any problems noted during the test and any problems found in analysis will be documented with a SPR form. If there is any question regarding the severity of the problem, the SQA representative will be the final judge.

The results of the post test briefing will be written up by the test team and reviewed by the SQA representative.

4.5.1.5 Installation and Checkout

This section should describe the SQA procedures to be used in monitoring the software installation and checkout activity.

The following italicized text offers an example for this section:

Software which passes formal testing is ready for installation at the customer's site. The SQA representative will ensure that all schedules for new or modified hardware have been met and that new or modified hardware has been tested and is ready to accept the tested software. This will be a cold start situation. After the system has been brought up, the tested software is loaded and a rerun of the test is conducted. The results of this rerun will be compared to the original test to determine if the system is ready for operation.

4.5.1.4 Verification and Validation

This section should describe the activities required for test planning, performance, reporting, and the organization units responsible for the activities. For projects where independent verification and validation is required, the verification and validation activities shall be the responsibility of an organization outside of the software development unit reporting directly to the (Acquirer).

The following italicized text offers an example for this section:

Software verification and validation is the process of ensuring that software being developed or changed will satisfy functional and other requirements (validation) and each step in the process of building the software yields the right products (verification). The SQA representative shall validate test procedures to specified requirements, verify the test plan implementation, witness performance tests, and have the capability to modify the test plan for special customer testing.

4.6 Software Acceptance and Delivery Phase

This section should describe the SQA procedures to be used when the software is ready for the acceptance and delivery phase.

The following italicized text offers an example for this section:

*Prior to the release of a deliverable (software or document), it must meet the approval of the **(XYZ Project Manager)** and the leader of the **(XYZ Project)** SQA representative. The necessary approval is signified by including a block of signatures on the cover sheet of the Release document, or on the software delivery cover letter.*

*The procedure leading up to the sign-off on the deliverable by the leader of the **(XYZ Project)** SQA representative, includes having passed all of the SQA milestones during the life-cycle steps leading up to the delivery. In addition to the basic approval steps, the leader of the **(XYZ Project)** SQA representative performs his own final check of the deliverable. These steps are documented below:*

(1) *Verify that all SQA documents or code were successfully completed, and that required signatures have been obtained.*

(2) *Assess the status of outstanding problems to determine if they are all closed and that corrective actions have been implemented and will be completed before acceptance. Deliverables may be released with outstanding problems only in an emergency.*

(3) *Perform an independent assessment of the deliverable (read the document or exercise the software) to form an opinion as to the quality level and the **(Organization)** image which will result from the release of the deliverable.*

(4) *If a hold on the deliverable is warranted, meet with the **(XYZ Project Manager)** to discuss the criticality of the timing of the release and the quality of the product to be released. Decide, based upon this discussion, whether or not to sign off on the delivery.*

(5) *If an impasse is reached in the decision whether or not to authorize the delivery, document this fact as an Issue, and notify the next higher level of **(Organization)** management.*

(6) *If the decision by the next higher level of management is to make the delivery, the delivery sign off on the "SQA Manager" line of the cover sheet shall be co-signed by the manager making the decision.*

4.7 Software Sustaining Engineering and Operations Phase

4.7.1 Delivery and Installation

This section should describe the SQA activities during the delivery and installation of the XYZ project.

The following italicized text offers an example for this section:

The SQA representative shall confirm that all copies are identical to the copy approved as a part of the Acceptance Phase. The SQA representative shall verify that the installation instructions that accompany the delivery are correct and complete.

4.7.2 Maintenance

This section should describe the SQA activities during the maintenance of the XYZ project.

The following italicized text offers an example for this section:

The SQA representative will continue to monitor the maintenance of the ***(XYZ Project)*** *software products after delivery to make certain that the quality (and particularly reliability) do not degrade as a result of maintenance actions. In monitoring the maintenance actions, the SQA representative will follow the procedures described in Section 3 of this plan. The SQA records shall continue to be maintained and updated during the SQA support of the maintenance activity. During the maintenance activity particular attention will be paid to the continuing operation of the problem reporting and corrective action system and making certain the software configuration management procedures are still followed.*

4.7.3 Additional Phases / Reviews

This section should include any additional reviews. Discuss each review in a separate section.

5. PROVIDER/SUBCONTRACTOR CONTROL

This section should describe the SQA procedures to evaluate provider/subcontractor products and commercially available, reusable, and government furnished software.

5.1 Sub-supplier Products

This section should describe the plans for evaluating the quality of the provider/subcontractor products prior to acceptance.

The following italicized text offers an example for this section:

A provider/subcontractor is any outside organization developing computer program products and services for the prime contractor. The SQA responsibilities for this area shall include:

- *Identification of contract quality requirements to ensure inclusion within provider/subcontractor's Request For Proposals and Request For Quotations*
- *Provider/subcontractor's surveys, inspections, and audits*
- *Evaluation of provider/subcontractor's system of quality assurance and control*
- *Review provider/subcontractor's deliverable requirements list documentation*
- *Audits of formal design reviews, acceptance testing, and configuration audits conducted by the provider/subcontractor*
- *Review contract changes/amendments to assure inclusion of or consistency with quality requirements.*

5.2 Commercially Available, Reusable, and Government Furnished Software

This section should describe the plans for evaluation of the reliability and quality of commercially available, reusable, and Government Furnished Software.

6. ACRONYMS / GLOSSARY

This section should contain an alphabetical list and definitions of all acronyms and abbreviations used in the document, all proper nouns, and any word used in a non-standard way. The acronym list includes, but are not limited to, the following:

CDR	Critical Design Review
CMM	Capability Maturity Model
ISO	International Organization for Standardization
LaRC	Langley Research Center
LHB	Langley Handbook
NASA	National Aeronautics and Space Administration
PDR	Preliminary Design Review

PM	Project Manager
SCM	Software Configuration Management
SCMP	Software Configuration Management Plan
SDP	Software Development Plan
SEI	Software Engineering Institute
SPR	System/Software Problem Report
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan
SRR	Software Requirements Review
STD	Standard
STPP	Software Test Plan and Procedures

7. CHECKLISTS/TOOLS

This section should include the specific checklists and tools located in the Software Program Assurance LHB Chapters 4 and 5, as well as any additional project specific tools or checklists that will apply and be used for software assurance for the XYZ Project.

APPENDICES

Any figures, forms, tables, or checklists that appear to be out-of-place in the text or cover more than a single page, should be attached as appendices of the SQAP. For example, the organizational chart including the XYZ Project team, the major milestones of the XYZ project including the SQA activities, the SPR form, etc. The following page shows examples for two of the above appendices:

Appendix A

XYZ Project -- Team Organization Chart

Name	Division	Role

Appendix B

XYZ Project Software Milestone Schedule

MILESTONES:

Task Name	Start	Finish
Develop Software Management Plan		
Develop Software Requirements Document		
Prototype & Support Software		
Develop Product Specification		
Develop Code		
Integration & Test – Software		
Integration & Test – Hardware		

SQA Activities:

Develop Software Quality Assurance Plan
Software Concept and Initiation Phase
Software Requirements Phase
Software Architectural (Preliminary) Design Phase
Software Detailed Design Phase
Software Implementation Phase
Software Integration and Test Phase
Software Acceptance and Delivery Phase
Software Sustaining Engineering and Operations Phase

**** End of Template ****

Appendix C: Software Quality Assurance Checklists

This section contains checklists to be used when evaluating the software quality assurance of a specific project. All, some, or none of these checklists may be used depending on the specific project.

In preparing a specific SQAP, these checklists shall be tailored by the acquirer (e.g., LaRC program/project manager) in accordance with the software project. These checklists are a starting point not the final answer in evaluating software quality assurance activities.

TABLE 1 Software Quality Evaluation Checklist

Reviewer ________________________________ Project ____
Date(s) of Review __________________________ Products Examined ______________________________

		YES	NO	N/A	NOTES
1.	Has the Software Quality Assurance Plan (SQAP) been updated to make it relevant to the current status and activities of the project?				
2.	Is the SQAP being followed?				
3.	Is the project on schedule and are the SQA activities up-to-date with this schedule progress?				
4.	Is the project following corporate SQA standards and procedures?				
5.	Is the project following the additional standards and procedures required by the customer and application area?				
6.	Have regular and periodic reports about the projects SQA activities and accomplishments been submitted to the corporate SQA manager?				
7.	Are records of results of all SQA evaluations being kept in the project's SQA files?				
8.	According to the records in the SQA files, have corrective actions been processed in a timely manner?				
9.	Are all checklist negative answers corrected or noted in documented issues?				
10.	Have all internal reviews been held as scheduled and have records of these reviews been filed?				
11.	Have all reviews of support functions (Software Configuration Management, Software Development Library, etc.) been held? Are support functions operating effectively?				
12.	Are problem reports being completely and correctly filled out?				
13.	Are follow-up corrective actions being documented and implemented?				
14.	Are impacts/side effects of corrective changes being investigated and resolved (impact on other modules, user documentation, etc.)?				
15.	Are the causes of problems being traced to their root causes? Are the development and SQA processes being changed to eliminate these root causes?				
16.	Are the data for software metrics being recorded and evaluated?				

TABLE 1 Software Quality Evaluation Checklist (cont.)

Reviewer ________________________________ Project ______________________________________
Date(s) of Review _________________________ Products Examined ____________________________

		YES	NO	N/A	NOTES
17.	Are the sign-off procedures prior to delivery being followed? Have any products been released or delivered without SQA approval or over the objections of SQA?				
18.	Does the project SQA representative meet frequently with software development leaders to discuss plans, accomplishments, problems and concerns?				
19.	Does the SQA representative leader report frequently to the project manager regarding status and problems of the software development process and products?				

TABLE 2 Software Configuration Management Review Checklist

Reviewer ________________________________ Project ______________________________________
Date(s) of Review _________________________ Products Examined _____________________________

		YES	NO	N/A	NOTES
1.	Has the Software Configuration Control Board (SCCB) met regularly?				
2.	Do SCCB members constitute all disciplines of the project organization?				
3.	Is the Requirements Baseline (RBL) under formal Software Configuration Management (SCM) baseline control?				
4.	Does the RBL have approval status?				
5.	Are all updates and changes to the RBL approved and accounted for?				
6.	Is the Functional Baseline (FBL) under formal SCM baseline control?				
7.	Does the FBL have approval status?				
8.	Are all updates and changes to the FBL approved and accounted for?				
9.	Is the Allocated Baseline (ABL) under formal SCM baseline control?				
10.	Does the ABL have approval status?				
11.	Are all updates and changes to the ABL approved and accounted for?				
12.	Is the Developmental Baseline (DBL) under formal SCM baseline control?				
13.	Does the DBL have approval status?				
14.	Are all updates and changes to the DBL approved and accounted for?				
15.	Is the Product Baseline (PBL) under formal SCM baseline control?				
16.	Does the PBL have approval status?				
17.	Are all updates and changes to the PBL approved and accounted for?				
18.	Is there an up-to-date list of all units in a software component?				
19.	For all software units required to support a given test:				
a.	Are all required units uniquely identified and uniquely name/numbered?				
b.	Do all required units have updated program listings with unique identifiers?				
c.	Are unit test results baselined?				

TABLE 2 Software Configuration Management Review Checklist (cont.)

Reviewer ______________________ Project ______________________
Date(s) of Review ______________________ Products Examined ______________________

		YES	NO	N/A	NOTES
20.	For the group of SQA approved units which have been integrated together into a build to support integration testing:				
a.	Has the build been baselined prior to test?				
b.	Are problems (encountered during the test) uniquely identified and accurately recorded against the current build?				
c.	Are test plans reviewed, approved, and baselined prior to test?				
d.	Are test procedures reviewed, approved, and baselined prior to test?				
21.	For formal, deliverable Product Baselines:				
a.	Are quality concerns satisfied for the deliverable version of the Functional Baseline Products?				
b.	Are quality concerns satisfied for the deliverable version of all baselined products?				
c.	Is all software baselined?				
d.	Are all version and revision identifiers unique and current?				
e.	Are all test plans and procedures updated and approved?				
f.	Are all test results certified by SQA as accurate findings of the tests?				
g.	Are all problem reports and changes approved and closed out?				
h.	Have all waivers or deviations been accepted by the customer?				
22.	Are there any out-of-date files in the SCM system?				
23.	Have any files been reserved for an unusually long period of time?				
24.	Has the Software Configuration Management Plan (SCMP) been kept up-to-date?				
25.	Are copyright statements included in each component?				
26.	Is corresponding source and executable code archived?				
27.	Is an up-to-date master list maintained of all archived media?				
28.	Is an up-to-date compilation list maintained for each software components?				
29.	Are all interface documents updated and approved?				

TABLE 3 Software Development Library (SDL) Review Checklist

Reviewer ______________________________ Project ______________________________
Date(s) of Review ______________________________ Products Examined ______________________________

		YES	NO	N/A	NOTES
1.	Is there a recognized approved SDL Control Procedure in the Configuration Management Plan?				
2.	Does the SDL control procedure discuss how the library function coordinates with, augments, and compliments (without being redundant) the function of CM?				
3.	Does the SDL control procedure define what program materials are to be entered into the SDL for this particular project?				
4.	Has SDL identification of library materials been carried out in compliance with CM practices?				
5.	Has SDL status accounting of library materials been carried out in compliance with CM practices?				
6.	Does the SDL differentiate and separate (in physical storage) software under development and software under formal baseline control?				
7.	Has the mechanism for placing all materials into the library (review, signature approval, and acceptance) been followed?				
8.	Is there a complete up-to-date status log of all materials contained in the library?				
9.	Is there a complete up-to-date status log of all approved changes to library materials?				
10.	Are all library materials (master tapes, discs, cards, listing documents, logs, etc.) protected from inadvertent or unauthorized use?				
11.	Are all library materials protected from inadvertent unauthorized change?				
12.	Are copies verified when made? (e.g., Is a comparator tool used for verifying the accuracy of software copies?)				
13.	Has the mechanism for releasing materials from the library (signature approval and release) been followed?				

TABLE 3 Software Development Library (SDL) Review Checklist (cont.)

Reviewer ______________________________ Project ____________________________________
Date(s) of Review ________________________ Products Examined ___________________________

		YES	NO	N/A	NOTES
14.	Are library logs maintained to assure a change against baselined software results in corresponding changes to affected documents and specifications?				
15.	Are library logs maintained to assure a change against a baselined document results in corresponding changes to software and other affected documents and specifications?				

TABLE 4 Corrective Action System Evaluation Checklist

Reviewer ______________________________ Project ____________________________________
Date(s) of Review ________________________ Products Examined ___________________________

		YES	NO	N/A	NOTES
1.	Are problem reporting procedures clearly documented and usable?				
2.	Are problem reporting procedures distributed to all staff who need to use them?				
3.	Is there an analysis procedure and is this procedure adequately staffed?				
4.	Is there a documented problem report categorization and prioritization procedure?				
5.	Is there a procedure for analysis of problem trends? Is this procedure followed?				
6.	Is there a documented procedure for submitting and following up on recommended corrective actions? Does this procedure include proper authorization of actions? Is this procedure followed?				
7.	Is there a thorough and complete documentation and record-keeping procedure for problems and corrective actions?				
8.	Have evaluations been conducted of the effectiveness of corrective actions after they have been taken?				
9.	Is there a documented procedure, with proper authorization, for closing out completed corrective actions?				
10.	Have plans been made to facilitate customer visibility into the corrective action system?				
11.	Are there procedures for identifying problems of traceability between the requirements and preliminary design? Preliminary design and detailed design? Detailed design and code? Are these procedures followed?				

TABLE 5 Hardware - Software Allocation Checklist

Reviewer ____________________ Project ____________________
Date(s) of Review ____________________ Products Examined ____________________

		YES	NO	N/A	NOTES
1.	Have interface specifications been agreed upon by those departments/organizations with software and hardware responsibilities?				
2.	Is new special-to-type hardware a feature of the project?				
3.	Is the special hardware the processor and /or its memory?				
4.	Are 'power-up' and power-fail' procedures specified and agreed upon?				
5.	Have sufficient software tests, using simulation or modeling techniques where required, been devised at various levels to exercise the software throughout its progressive development?				
6.	Are special hardware facilities required to simulate eventual hardware and allow software proving?				
7.	Where the new hardware is functionally similar to previous items, has consideration been given to common input/output codes?				
8.	Are system start-up delays required? If so, are they hardware or software implementation and can they be over-ridden: (a) in a functioning system? (b) during test to permit fault finding?				
9.	Are occasional input/output errors permitted, detectable, logged, and recoverable?				
10.	Will it be possible to integrate and test hardware and software sub-assemblies prior to full system integration?				
11.	Are separate integration specifications required, available, and agreed upon?				
12.	Are responsibilities for monitoring and documenting integration defined?				
13.	Is hardware or software data buffering employed across the interfaces? If so, is overflow action specified and agreed upon?				
14.	Do standard driver routines exist for this hardware?				

TABLE 5 Hardware - Software Allocation Checklist (cont.)

Reviewer ___________________________________ Project ____
Date(s) of Review ____________________________ Products Examined ________________________________

		YES	NO	N/A	NOTES
15.	If special drivers are required are they to exist at the operating system level and conform to its rules?				
16.	How far will software integrity be established against the full speed range of hardware timing possibilities?				
17.	Is special software required to verify hardware and interface design?				
18.	Are special hardware checking routines required: (a) as part of the ‘normal’ system software? (b) for periodic maintenance or servicing?				
19.	Are the special hardware checking routines written, tested and agreed by the responsible hardware and software leaders?				
20.	If system software detects a hardware malfunction, can it: (a) use other hardware in a reconfigured mode? (b) notify or record the malfunction?				
21.	Can hardware faults be simulated to check (a) and (b) above?				
22.	Is special test equipment required and available for interface debugging?				

TABLE 6 Software Requirements Document Review Checklist

Reviewer ________________________________ Project ____
Date(s) of Review __________________________ Products Examined ______________________________

		YES	NO	N/A	NOTES
1.	Is the system's functional flow clearly and completely described?				
2.	Has each decision, selection and computational function that the software must perform been clearly defined?				
3.	Is there a complete definition of software functions requiring human reactions and operator response?				
4.	Is there a description of the performance and capacities required of each function?				
5.	Does each software function that is specified trace to one or more system requirements?				
6.	Is there a description of the inputs and outputs of each function?				
7.	Can the required performance and capabilities be achieved?				
8.	Are all common functions identified and appropriate linkage mechanisms enabling their common use defined?				
9.	Is a dictionary for all data elements provided and is it complete?				
10.	Do the input descriptions match the output descriptions for inter-functional communications?				
11.	Are timing requirements given?				
12.	Are memory requirements given?				
13.	Are the timing and memory limits compatible with hardware constraints and with system timing and capacity budgets?				
14.	Are all limits and restrictions on software performance defined?				
15.	Is there a description of the executive system requirements?				
16.	Are the support software provisions and limitations given?				
17.	Is the method for demonstrating compliance with requirements defined and are all limitations of these methods explained?				
18.	Are standards and naming conventions established for the project followed?				
19.	Are all functions testable?				
20.	Are required software quality attributes (e.g., reliability, portability, reusability, maintainability, etc.) defined?				

TABLE 7 Software Design Document Review Checklist

Reviewer ______________________________ Project ______________________________
Date(s) of Review ______________________________ Products Examined ______________________________

		YES	NO	N/A	NOTES
1.	Is the structure of the software compatible with the functional requirements and with sound design practices?				
2.	Is there a complete listing of all symbols and definitions?				
3.	Is there a complete listing of all symbol meanings and parameter values?				
4.	Does each unit description include a statement of the functions to be performed by that unit?				
5.	Is there a unit or combination of units to perform every required function?				
6.	Does functional media show the relationships, both data and control, between all units?				
7.	Are all conditions of system status that call for a unit response described?				
8.	Is the flow of input and computed output shown?				
9.	Do the input descriptions match the output descriptions of the unit generating that output?				
10.	Are the specifications of all interfaces with the software executive shown?				
11.	Are the algorithms or equations to be evaluated by each unit completely and correctly presented?				
12.	Are the entry and exit points of each unit listed?				
13.	Are all units that may be called by this unit listed?				
14.	Are all the units that may call this unit listed?				
15.	Has the logic within each unit been given and the decision criteria been defined, whether system status, operational mode, or computed?				
16.	Is there a description of software error conditions, error exits, and recovery procedures?				
17.	Is the design of sufficient detail to allow coding?				

TABLE 7 Software Design Document Review Checklist (cont.)

Reviewer ______________________________ Project ____________________________________
Date(s) of Review __________________________ Products Examined ____________________________

		YES	NO	N/A	NOTES
18.	Can the units be tested as designed?				
19.	Is the unit's functional flow clearly described?				
20.	Are all common functions identified and appropriate linkage mechanisms enabling the sharing of these functions defined?				
21.	Are specified buffer sizes adequate?				
22.	Are there adequate margins for minimum/maximum data values?				
23.	Have design conventions and standards been followed?				
24.	Is the functional role of each table and block clearly defined?				
25.	Do these data objects fulfill these stated functions?				
26.	Is the scope of each table and block clearly defined?				
27.	Is each table's and block's structure clearly described, giving a pictorial representation, and description of each field's name, size, location, contents, and level?				
28.	Is there a list of all units reading from, or writing into, each table or block?				
29.	Is each table organized in a logical and efficient manner, consistent with its use by various units at different levels in the program hierarchy?				
30.	Is there a correct description of the input from each unit including its type, source, format, units, accuracy, and quantity?				
31.	Is there a correct description of the output from each unit, including its type, destination, format, units, accuracy, and quantity?				
32.	Is undesirable underflow or overflow that is undetected avoided?				
33.	Are mathematical algorithms properly implemented?				
34.	Are interrupts properly handled?				
35.	Are all referenced data items declared?				
36.	Is the use of each data item valid (based on its type designation)?				
37.	Are all declared data items actually used in this unit?				

TABLE 7 Software Design Document Review Checklist (cont.)

Reviewer ____________________________ Project ____________________________
Date(s) of Review ____________________________ Products Examined ____________________________

		YES	NO	N/A	NOTES
38.	Are the data items used only after they have been assigned values?				
39.	Do all of the unit's possible control paths terminate? (i.e., "no endless loops")?				
40.	Are all the calls made by this unit shown in the calling hierarchy?				
41.	Do all calls made by this unit follow the calling conventions of the programming guidelines?				
42.	Do all names defined in this unit follow the conventions of the programming guidelines?				
43.	Does the unit's control logic flow from top to bottom?				
44.	Does this unit have a single entrance and a single exit?				
45.	Does this unit determine the values of all inputs passed to units it calls?				
46.	Does this unit use all outputs returned by units it calls?				
47.	Does this unit avoid the use of function or control codes to determine the function or flow of control of units it calls?				
48.	Does this unit avoid using external data items that are used by other units in a manner that violates programming standards?				

TABLE 8 Software Code Review Checklist

Reviewer ________________________________ Project ________________________________
Date(s) of Review ________________________________ Products Examined ________________________________

		YES	NO	N/A	NOTES
1.	Is unexecutable code avoided?				
2.	Are all declared data items actually used?				
3.	Is redundant code avoided?				
4.	Are endless loops avoided?				
5.	Are references to undefined data items (before assignment) avoided?				
6.	Is argument passing compatible?				
7.	Do all branches point to correct locations?				
8.	Are all alternatives of conditional execution correct?				
9.	Is initialization for all program states correct?				
10.	Is common storage protected?				
11.	Are constants defined with the correct values?				
12.	Are mathematical algorithms properly implemented?				
13.	Are buffer sizes adequate?				
14.	Are margins for minimum/maximum data values adequate?				
15.	Is the program logic correct?				
16.	Are comments clear, concise, consistent, and sufficient and follow the standards?				
17.	Have coding standards and conventions been followed?				

TABLE 9 Software Test Plan Review Checklist

Reviewer ________________________________ Project ____
Date(s) of Review __________________________ Products Examined ______________________________

		YES	NO	N/A	NOTES
1.	Is the location and schedule for the testing given?				
2.	Are the responsibilities of each of the test personnel clearly delineated?				
3.	Are the objectives of the test effort described and are they compatible with program objectives?				
4.	Will each requirement be demonstrated as fulfilled by means of the test effort?				
5.	Does all testing to be done aid in demonstrating compliance with the defined requirements?				
6.	Does the proposed test effort correspond to the structure of both the requirements and the software?				
7.	Are all applicable documents, user’s manuals and programmer guides referenced?				
8.	Are methods given for evaluating whether units correspond to individual unit specifications?				
9.	Are limitations of test methods and facilities described, and are adequate steps taken to minimize the effort of those limitations?				
10.	Is provision made for the review and verification of test input data?				
11.	Are the general configurations of the software and test environment described?				
12.	Is the conduct of the tests described?				
13.	Are there procedures for the configuration control of test tools and facilities?				
14.	Are there procedures for the configuration control of software elements under test?				
15.	Is adequate provision made for regression testing?				
16.	Is the content of the test reports described and will the planned tests contribute the information needed to complete the test reports?				

TABLE 9 Software Test Plan Review Checklist (cont.)

Reviewer ______________________ Project ______________________
Date(s) of Review ______________________ Products Examined ______________________

		YES	NO	N/A	NOTES
17.	Is the anticipated test coverage given and is this coverage adequate?				
18.	Are requirements for special data reduction and analysis efforts described?				
19.	Does the test effort make adequate provision for the control and incorporation of changes to the specification, design, or code that may occur during the test effort?				

TABLE 10 Software Test Procedure Review Checklist

Reviewer ______________________________ Project ____________________________________
Date(s) of Review ________________________ Products Examined ___________________________

		YES	NO	N/A	NOTES
1.	Is each test specified in the test specifications covered?				
2.	Does each procedure completely describe instructions for conducting the test?				
3.	Does the test suite complement and extend the unit test suites? Is the correct test environment being considered for each type of test?				
4.	Are sufficient test sets used in each type of testing?				
5.	Is there criteria for determining which tests will be run?				
6.	Are all tests and supporting software under configuration control?				
7.	Is all software to be tested under configuration control?				
8.	Are testing schedules established and followed?				
9.	Is there tracking of daily, weekly, and cumulative test results?				
10.	Are backup tapes maintained for all testing baselines?				
11.	Are there sufficient procedures for regression testing?				

TABLE 11 Software Test Conduct Review Checklist

Reviewer ______________________ Project ______________________
Date(s) of Review ______________________ Products Examined ______________________

		YES	NO	N/A	NOTES
1.	Was the hardware and software configured as specified in the test procedure?				
2.	Was the global test environment set up as specified in the test procedure?				
3.	Were the test input data values those specified in the test procedures?				
4.	Was each test conducted as specified in the test procedure?				
5.	Were all tests performed in the sequence specified in the test plan?				
6.	Was each test recorded as specified in the test procedure?				
7.	Do test results correspond to expected test results?				
8.	Does the application of the test criteria indicate a correct result?				
9.	Can differences between the expected results and the observed results be reconciled?				
10.	Is additional testing required to isolate problems and trace them to their causes?				

Appendix D: Software Quality Assurance Tools

The following table contains a list of commercially available, off-the-shelf software tools. These tools automate many quality-related activities in the areas of information organization, software analysis, and software testing. These tools can be used with benefit by both software quality assurance teams and software developers.

This list of software tools is a sample of the many that are available today and will be available in the near future. A regular review of the technical literature and a frequent examination of software tool catalogs will help in keeping current in this important subject.

Tool Name	Tool Type (P)	Host	Language	Vendor	Abstract
AdaQuest™ Static Analyzer Tool	Code Analyzer		Ada	General Research Corporation	Detects logic errors, checks for standards violations, and analyzes branch and call structure
COBOL Glossary	Code Analyzer	IBM Mainframe	COBOL	MacKinney Systems	Analyzes data cross-references
FSCAN	Code Analyzer	Several	FORTRAN	International Logic Corporation	Provides editor, code analyzer and cross-reference checking features
ISAS	Code Analyzer	DEC & Data General	Many	Singer	Analyzes source code
Probe/38	Code Analyzer	IBM System 38	RPG, COBOL	Advanced System Concepts	Analyzes control flow cross-references
REFTRAN	Code Analyzer	Several	FORTRAN	William R. DeHaan	Checks consistency between cross-references
SCAN/COBOL	Code Analyzer	IBM MVS	COBOL	Computer Data Systems	Analyzes control flow
SCAN/COBOL	Code Analyzer	Several	COBOL	Group Operations, Inc.	Analyzes source code
CA-ACCUCHECK	Comparator	IBM Mainframe		Computer Associates International, Inc.	Compares files, records and fields
COMAREX	Comparator	IBM Mainframe		Sterling Software	Compares source, object code, JCL and files
Core Image Compare Program	Comparator	IBM Mainframe		Coopers and Lybrand	Compares object code to load modules
RES-Q	Comparator	IBM Mainframe		Quality Systems Development Corporation	Compares source code and updates
SHOWDIFF	Comparator	HP 3000		NOI Systems	Compares ASCII files

Tool Name	Tool Type (P)	Host	Language	Vendor	Abstract
Source Compare Program	Comparator	IBM Mainframe		Coopers and Lybrand	Compares two source code programs
Source Compare Program	Comparator	IBM Mainframe		MacKinney Systems	Compares two programs and JCL
SUPER-C	Comparator	IBM Mainframe		IBM	Utility for source code compare and auditing
PC-Metric™	Metrics	PC	Ada, assembler, C, C++, COBOL, dbase, FORTRAN, MODULA-2, Pascal	SET Laboratories, Inc.	Computes software science and cyclomatic complexity metrics; checks standards compliance
UX-Metric™	Metrics	UNIX	Ada, assembler, C, C++, COBOL, dbase, FORTRAN, MODULA-2, Pascal	SET Laboratories, Inc.	Computes software science and cyclomatic complexity metrics; checks standards compliance. Similar to PC-METRIC.
Inspector	Metrics	MVS, VM		KnowledgeWare	Computes quality metrics
PM/SS	Metrics	DOS, IBM MVS		Adpac Corp.	Measures and reports code maintainability.
Corrective Action 2.0	Problem Report Tracking			The Harrington Group, Inc.	Records problem report information, offers user access control, produces graphs of trends and Pareto analysis.
Defect Control System (DCS)	Problem Report Tracking			The Software Edge, Inc.	Supports defect tracking through defect entry, notification, classification and trend reporting. Access control provided.
QA Defect Tracking System	Problem Report Tracking			Honickman & Associates Limited	
SQA Manager	Problem Report Tracking	IBM PC or Compatible		Software Quality Automation	Collects problem reports and associated data; analyzes problem report data and forecasts test time and reliability.
SQMS/Testing™	Problem Report Tracking	Sun SPARCstation		Software Quality Tools Corporation	Tracks software problem report and size metrics
Automator qa	Regression Test	MVS, VM, VMS, UNIX, MPE, OS/400		Direct Technology	Records and playback test scripts; generates random tests given an array of possible input values.

Tool Name	Tool Type (P)	Host	Language	Vendor	Abstract
AutoTester	Regression Test	IBM PC or compatible		Software Recording Corporation	Provides capture, replay and test results comparison. Allows editing of test scripts.
Bloodhound	Regression Test	IBM PC or compatible		Goldbrick Software	Captures keystrokes and screen images, when scrolling occurs.
Bloodhound	Regression Test	IBM PC		Goldbrick Software	Records and plays back keystrokes and screens in text mode for re-testing.
TestPro™	Regression Test	Windows		Sterling Software	Records and plays back keystrokes and mouse actions for re-testing, provides screen-save and test comparison capabilities.
CapBak	Regression Test	IBM PC, UNIX		Software Research, Inc.	Records and plays back keystrokes for re-testing, provides screen-save and save editing capabilities.
SQA:Robot	Regression Test	Windows, OS/2		Software Quality Automation	Records test actions in GUI environment; plays back all actions. Allows editing of scripts.
ACT	Standards Auditor		Ada	TEXEL & Co., Inc.	Checks for compliance with Ada coding standards.
AdaAssured™	Standards Auditor	DECstation, IBM RS 6000, HP 9000, Sun SPARCstation	Ada	Gramma Tech	
C Portability Verifier	Standards Auditor	UNIX	C	Mindcraft, Inc.	Checks for standards compliance and for portability between C compilers
CodeCheck™	Standards Auditor	DOS, Macintosh, OS/2	C, C++	Abraxas™ Software, Inc.	Checks code for standards, portability compliance; computes Halstead and complexity metrics
Enforcer II	Standards Auditor		COBOL	Clarity Concept Systems	Checks for compliance with coding standards
INSPECTOR	Standards Auditor	IBM MVS or VM	COBOL	Language Technology	Checks for standards compliance and computes metrics
FORTRAN-lint	Static Analyzer	DEC VAX, Data General MV	FORTRAN	Information Processing Techniques, Inc.	Detects coding errors such as use of variables before declaration, unused declared variables, type mismatches, etc.

Tool Name	Tool Type (P)	Host	Language	Vendor	Abstract
QA Partner	Test Analysis	Windows, Macintosh, Motif, Open Look		Segue Software	Tests GUIs using object comparisons
Ada Test and Verification System (ATVS)	Test Coverage Analyzer	DEC VAX	Ada	General Research Corporation	Performs static analysis, identifies unreachable code and logic errors, audits for standards compliance, reports test coverage.
AdaQuest™ Dynamic Analyzer Tool	Test Coverage Analyzer		Ada	General Research Corporation	Checks for branches executed by tests, verifies interface and timing constraints, and checks executable assertions.
Analysis of Complexity Tool (ACT)	Test Coverage Analyzer	Several	Several	McCabe Associates	
Analyzer	Test Coverage Analyzer	HP 3000, IBM Mainframe	COBOL	ALDON Corporation	
CHECKOUT	Test Coverage Analyzer	Unisys	COBOL	Programming Aids, Inc.	
Logiscope	Test Coverage Analyzer	Several	Several	Verilog	
Path Analysis Tool (PAT)	Test Coverage Analyzer	DEC VAX, IBM, HP, SUN, Apollo, PC, Mac	FORTRAN	Science Applications International, Corp.	Provides code execution coverage for a test or set of tests; code execution frequencies for modules and paths are reported.
Performance and Coverage Analyzer	Test Coverage Analyzer	DEC VAX	Several	Digital Equipment Corporation	
RXVP80	Test Coverage Analyzer	Several	FORTRAN	General Research Corporation	
TCAT/C	Test Coverage Analyzer	IBM, UNIX, DOS	COBOL, Pascal	Software Research Associates	
TMOON	Test Coverage Analyzer		Ada	DDC-I, Inc.	Provides statement and condition coverage, statement execution counts, and subprogram execution times.
AdaQuest™ Task Analyzer Tool	Test Data Analyzer		Ada	General Research Corporation	Creates timing diagrams to help diagnose tasking errors.

Tool Name	Tool Type (P)	Host	Language	Vendor	Abstract
START™	Test Data Preparation	Several		McCabe Associates	Generates test conditions from data flows; interfaces with CASE tools.
Bounds-Checker	Test Monitor	IBM PC or compatible DOS	C, C++, assembly	Nu-Mega Technologies, Inc.	Validates pointers. Checks for out-of-range variable access, overwrite by an external program and memory access outside segment.
C-Debug	Test Monitor	IBM PC or compatible DOS	C	Softran Corp.	Validate pointers. Checks out-of-range accessing of variables and memory access outside segment.
INTRCPT	Test Monitor	IBM PC or compatible		Hackensack	Monitors interrupt invocations and displays entry and exit parameters.
MemCheck	Test Monitor	IBM PC or compatible DOS	C, C++	StratosWare Corp.	Validates pointers. Checks out-of-range accessing of variables.
Periscope/EM	Test Monitor	IBM PC or compatible DOS		The Periscope Company, Inc.	Provides real-time hardware breakpoints and captures a buffer of CPU event information.
SafeWin	Test Monitor	IBM PC or compatible with Windows	C, C++	SeaBreeze Software Systems	Validates pointers. Checks memory access outside segment. Traps fatal Windows errors and object creation.
Soft-ICE	Test Monitor	IBM PC or compatible DOS		Nu-Mega Technologies, Inc.	Provides hardware-like capabilities to set breakpoints on I/O and memory access through software emulation.
Meta Test	Test Planning	UNIX, DOS		Software Research Associates	Assists in writing test plans
R Trace	Traceability Database	DEC VAX		NASTEC Corporation	Requirements traceability database and checking
Turbo Trace	Traceability Database	DEC VAX		Computer Science Innovations	Requirements traceability database and checking